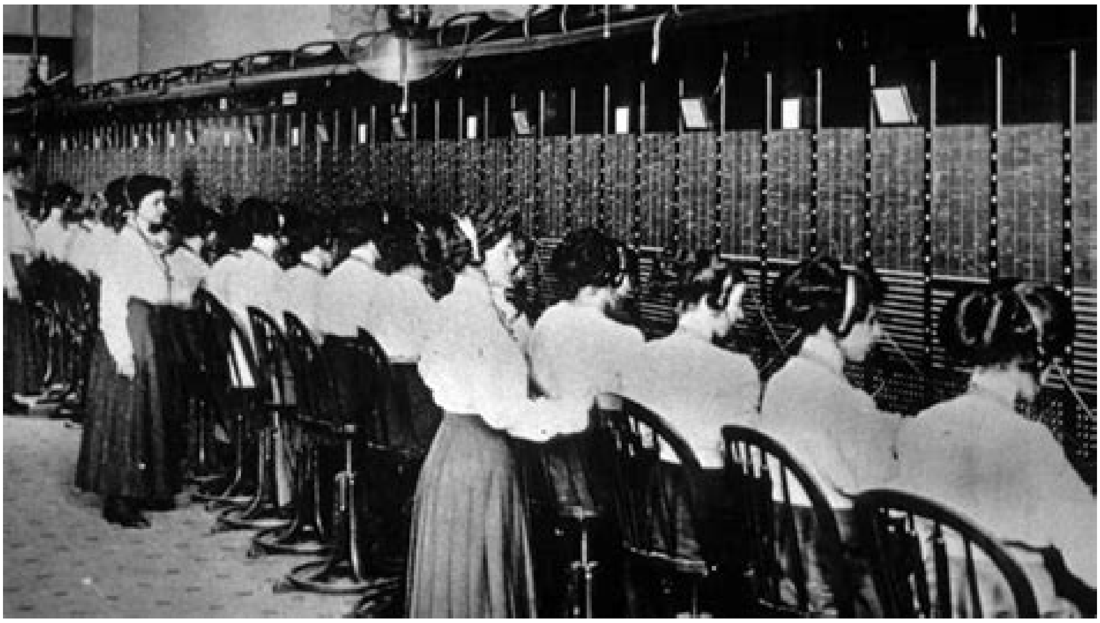
NEVARC NEWS VK3ANE

Incorporated in Victoria, 2014, Registration Number: A0061589C
The monthly magazine of the
North East Victoria Amateur Radio Club http://nevarc.org.au/
An Affiliated club of Wireless Institute of Australia
An Affiliated club of Radio Amateur Society of Australia Inc.

Volume No: 08 Issue 10 October 2021

Next Meeting
Sunday 10th October

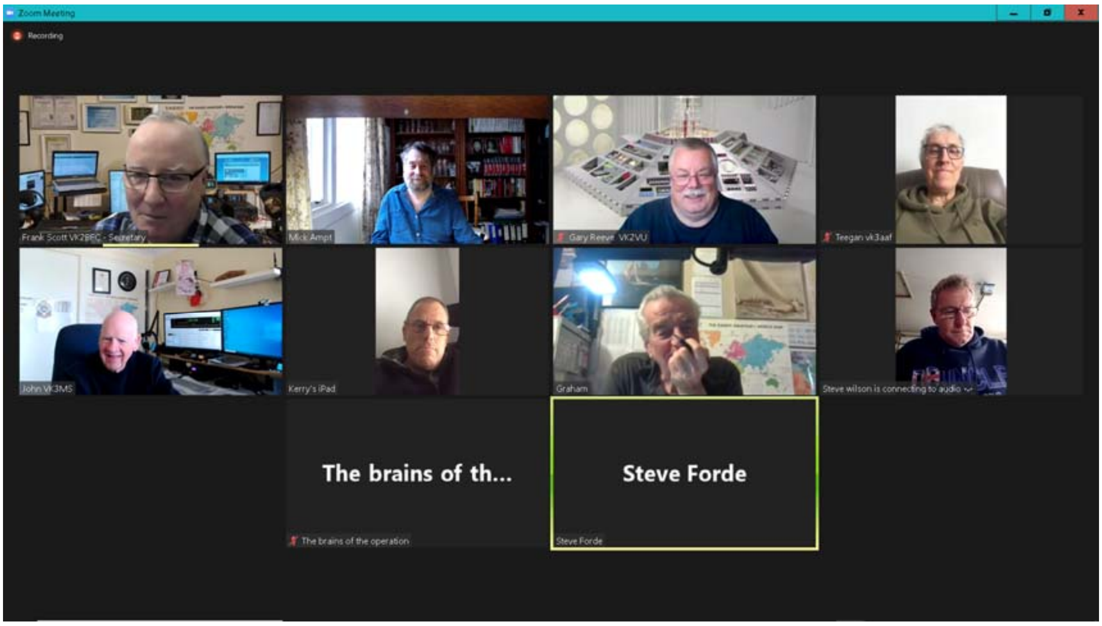
Due to changing COVID restrictions, often at short notice, details of the meeting will be emailed to members, so it may be either online or in person at the clubrooms



Dial in to NEVARC Repeaters, see page 4

NEVARC AGM

On Sunday, 12 September the online AGM was held.
All positions were declared vacant and all current incumbents offered themselves for re-election.
The club is in a very healthy financial position.
Despite COVID, the club has enquiries and increase in membership very month.
The club website has had improvements and an extensive club library upgrade, by Matt VK3VS.
There is over 6 gigabytes of articles in the library that members can download and read.
The club repeaters are all operating at peak capacity, including the Wires-X repeater.
The club has more Foundation licence classes planned for next year.



Committee of Management

Nominations were received for all vacant positions plus 2 committee positions.
As all positions were nominated unopposed the nominees were elected.

Nominees were:

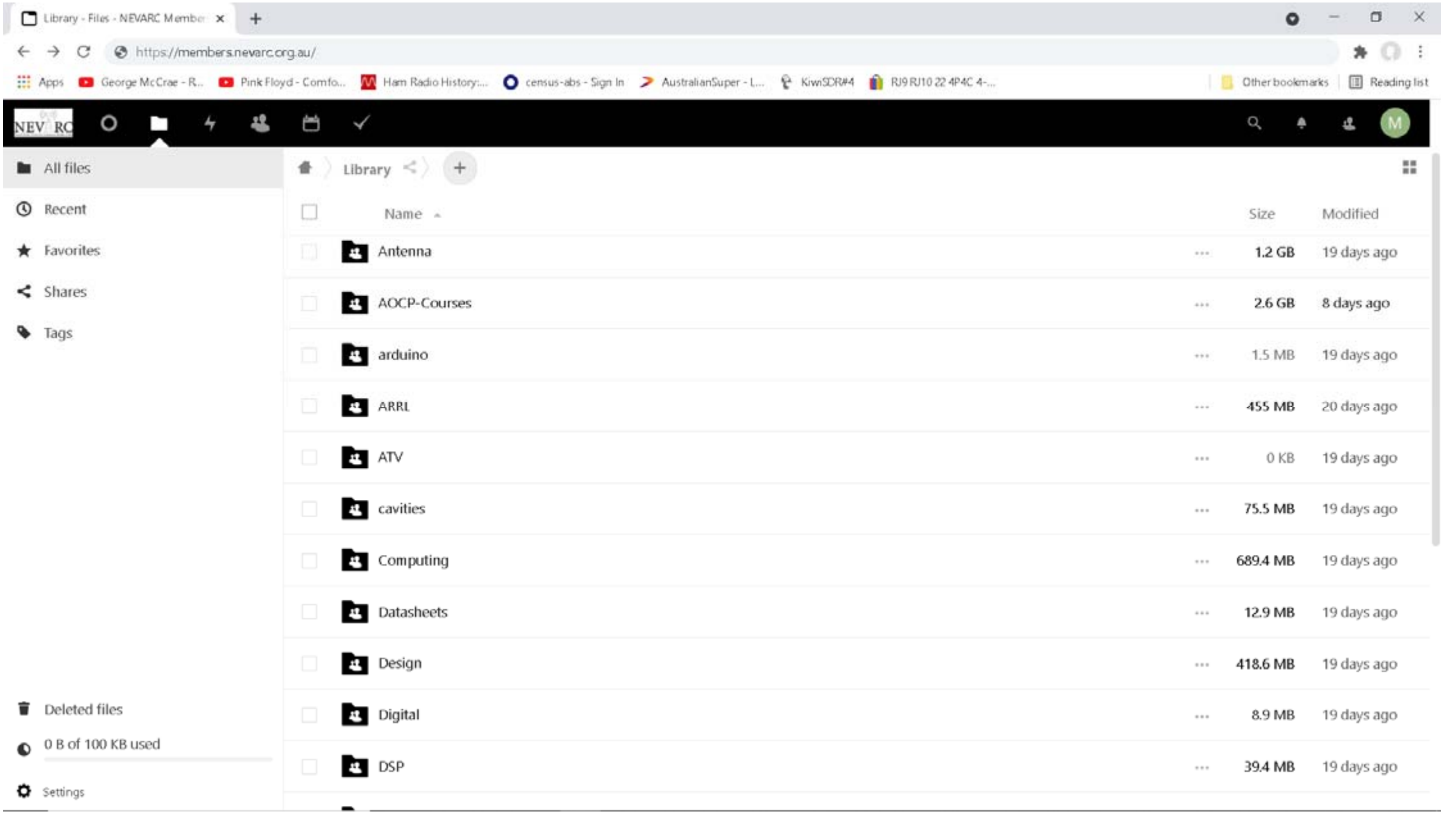
Matt Bilston VK3VS	– President
Gary Reeves VK2VU	– Vice President
Frank Scott VK2BFC	– Secretary
Amy Bilston	– Treasurer
Mick Ampt VK3CH	– Editor
Shane Carmison VK3KHS	– Committee
Graham Ayton VK2ER	– Committee

Advisory Committee

There are at present 2 advisory committees.
Contest committee – for the formation of groups and planning for club participation in national and international contests
Meetings committee – for the advancement of concepts and ideas directed at physical meetings of the club, in particular, technical presentations and demonstrations.

Expanded NEVARC website library for members

Matt VK3VS has been busy uploading heaps of files to read on the club member's area of the club website.
There is over 6GB of stuff to choose from.



A screen grab of just a small bit of the list available

Matt has scoured the internet, both public and the dark web, for articles of a ham radio interest.
The files are have been sorted into folders to make searching easier.

One file that took my curiosity was an article using an Oven Magnetron as a 13cm TX.
I would not be so crazy to attempt this, but if you are, well then this is for you.
This is just one of a whole list of files to read, easy to spend hours just looking.

There are study guides for ham radio, service manuals, technical white papers, SDR operation, the list goes on.

Just another bonus of membership with NEVARC…

NEVARC News
The club magazine
All it needs is YOU
Send stories of your radio news to the editor
What have you been up to in these strange days of COVID?
magazine@nevarc.org.au

Accessing NEVARC Repeaters via the Internet

THE IDEA

Matt gets things happening quickly.
At the AGM it was discussed how members living remotely from accessing NEVARC repeaters via RF, to be able to access, with authorised credentials, the club repeaters via the internet.
Less than 11 hours later, with a break for Matt VK3VS to, set the emails up for the new committee and executive, walk the dog, mow lawns, wash his car, cook a dinner in a BBQ smoker and relocate the smoker when the rain came, eat dinner, wash dishes and then chat to Mick VK3CH, then later on, Matt had Mick linked up to VK3RWO via both Mick's mobile phone and Mick's PC.

The program used [and there are many to choose from] selected by Matt was Zoiper. Zoiper is a software program for making telephone calls over the internet using a computer, via VoIP (Voice over IP)

TECHNICAL STUFF YOU DON'T NEED TO KNOW - SESSION INITIATION PROTOCOL (SIP)

The Session Initiation Protocol (SIP) is a signalling protocol used for initiating, maintaining, and terminating real-time sessions that include voice, video and messaging applications. SIP is used for signalling and controlling multimedia communication sessions in applications of Internet telephony for voice and video calls, in private IP telephone systems, in instant messaging over Internet Protocol (IP) networks as well as mobile phone calling over LTE (VoLTE).

The protocol defines the specific format of messages exchanged and the sequence of communications for cooperation of the participants. SIP is a text-based protocol, incorporating many elements of the Hypertext Transfer Protocol (HTTP) and the Simple Mail Transfer Protocol (SMTP). A call established with SIP may consist of multiple media streams, but no separate streams are required for applications, such as text messaging, that exchange data as payload in the SIP message.

SIP works in conjunction with several other protocols that specify and carry the session media.
Most commonly, media type and parameter negotiation and media setup are performed with the Session Description Protocol (SDP), which is carried as payload in SIP messages. SIP is designed to be independent of the underlying transport layer protocol and can be used with the User Datagram Protocol (UDP), the Transmission Control Protocol (TCP), and the Stream Control Transmission Protocol (SCTP). For secure transmissions of SIP messages over insecure network links, the protocol may be encrypted with Transport Layer Security (TLS). For the transmission of media streams (voice, video) the SDP payload carried in SIP messages typically employs the Real-time Transport Protocol (RTP) or the Secure Real-time Transport Protocol (SRTP).

SETTING UP ZOIPER5

Matt emailed me all the necessary instructions and software settings; my version is Zoiper5, which are;
Download a SIP phone app for your phone, or a Program for your desktop computer, https://www.zoiper.com
You will need these credentials, some or all will be needed:

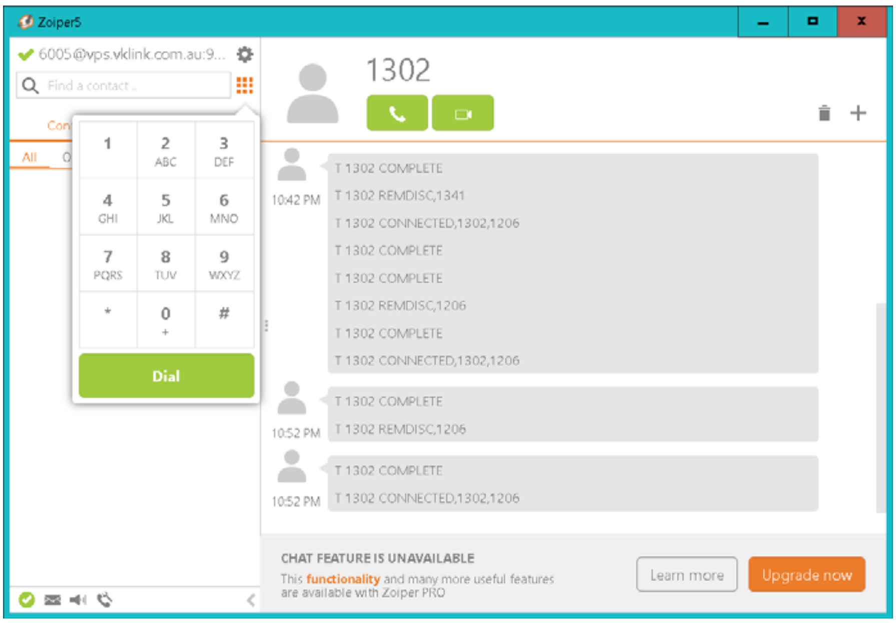
Username	[User Name Allocated by Matt]
Password	[Password allocated by Matt]
SIP port	9897
SIP domain	vps.vklink.com.au
Register port	9897
Expire time	600

SIP port and domain may be written like this: vps.vklink.com.au:9897
When that connects, you can dial:

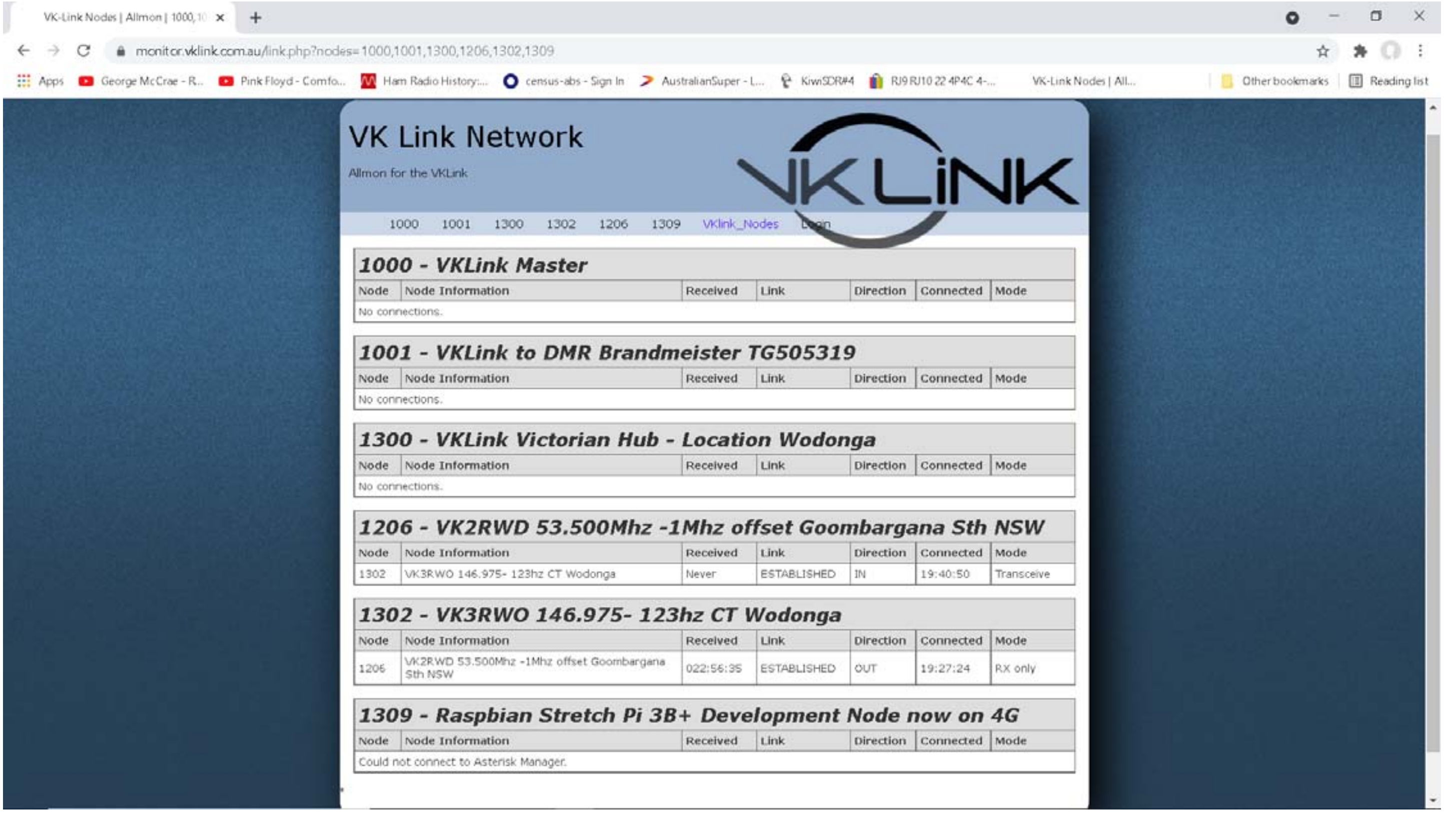
1302	Will bring you out on VK3RWO 2 Meter repeater (Except on Wednesday night where it will bring you out on VK2RWD)
1206	Will bring you out on VK2RWD 6 Meter repeater
6000 or 6001	Will dial Matt VK3VS IP phone
6004	Will dial John, VK3MS

I installed Zoiper5 on both my PC and iPhone. After some fiddling and guidance by Matt, it was going.
I was able to dial into my PC from my iPhone, or vice versa.
After some local testing I dialled into the VK2RWD 6 Meter repeater and could hear myself on my iPhone coming back on the repeater in Matt's shack.
You can also "see" your connection on a web browser at https://monitor.vklink.com.au/

The Zoiper5 screen



THE VKLINK MONITOR DISPLAYING STATUS OF NEVARC REPEATERS



With https://monitor.vklink.com.au/link.php?nodes=1000,1001,1300,1206,1302,1309 the connections to any of the linked repeaters can be seen in real time. Just select [Vklink_Nodes] from the menu.

USING ZOIPER5

To connect to the NEVARC repeater, start Zoiper5 and then “dial” the repeater of your choice.
So to get into VK3RWO 2 Meter repeater, dial 1302.
You need to wait a few seconds for all the packets to talk and link up.
To talk select the dial keypad and select * to key the repeater [talk] and # to un-key the repeater [listen]

It is important to leave a short break before talking to allow internet propagation of all the “smarts” before talking, like most internet chat programs.
I used it without any noticeable dramas and the audio is very good, not robotic at all.

As I have the PC, XYL Laptop, iPhone and a SIP Phone, Matt gave me accounts with four separate numbers;

6005 VK3CH Desktop PC
6006 VK3CH iPhone
6007 VK3CH Kitchen Laptop
6008 VK3CH SIP Phone

Remember:

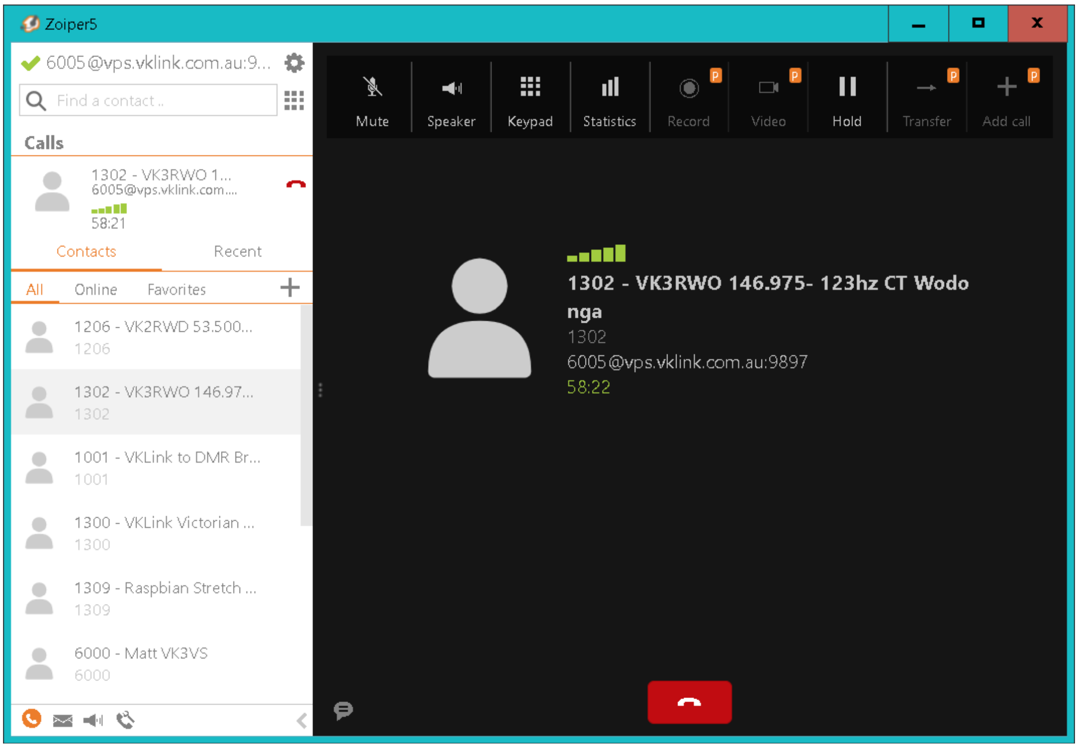
YOU NEED TO LEAVE A BREAK BETWEEN OVERS SO ROIP USERS CAN BREAK IN

LISTEN ANYTIME

Matt said you are able to dial up and just monitor the repeater anytime and just connect to talk when you hear someone come up. The PC would be better suited to that, but nothing stopping you using your iPhone, other than your battery life. If you use your iPhone on a WiFi connection then you don't use your data allocation. So using a PC, located in Melbourne, I may as well be in Wodonga or surrounds and chat like a local.

It is easy to forget you are on a repeater, so using proper amateur radio protocols of callsign identification and civil language are important, especially when using an iPhone, you often forget you are NOT on a phone call. The system has a one minute timeout, but it quickly resets, but still worth remembering.

There is no time limit on connecting in, so you can connect and listen for hours for any repeater traffic. Just like having a radio on the repeater frequency all day, waiting for activity.



Linked up to VK3RWO

DIAL IN TO NEVARC CLUB REPEATERS ON YOUR HOME TELEPHONE

Matt told me that I can even re-configure the telephony in my NBN router to be used... pick up your home phone and dial into VK3RWO 2 Meter repeater. Dial plans could be set in your router that if it dials a 4 digit number starting with 1; it goes to the VKLink server instead of your NBN provider. Absolutely anything is possible with asterisk/sip.

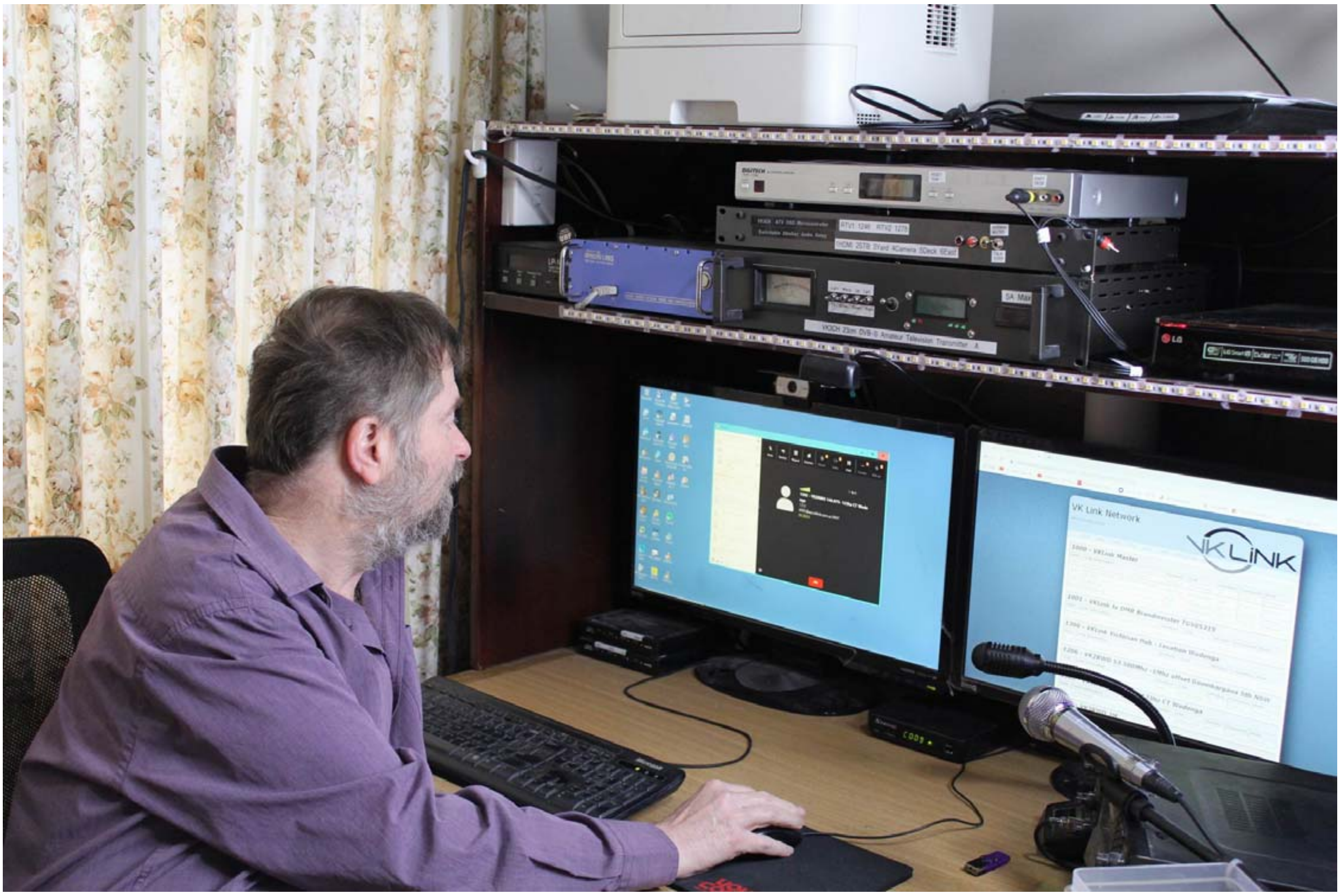
The steps are to, Log into your router, Go into telephony, Edit settings or make another profile.

When I tried to set it up, the Optus [locked down] router did not have the Telephony option available, because they want to charge you for phone calls, not have you calling people via VoIP for free!

You can always buy another router and given how quickly they become outdated due to security flaws, it is probably a good idea. The PC and iPhone will do me for now.

THE REAL TEST - JOINING THE WEDNESDAY NIGHT 8.00 PM NET

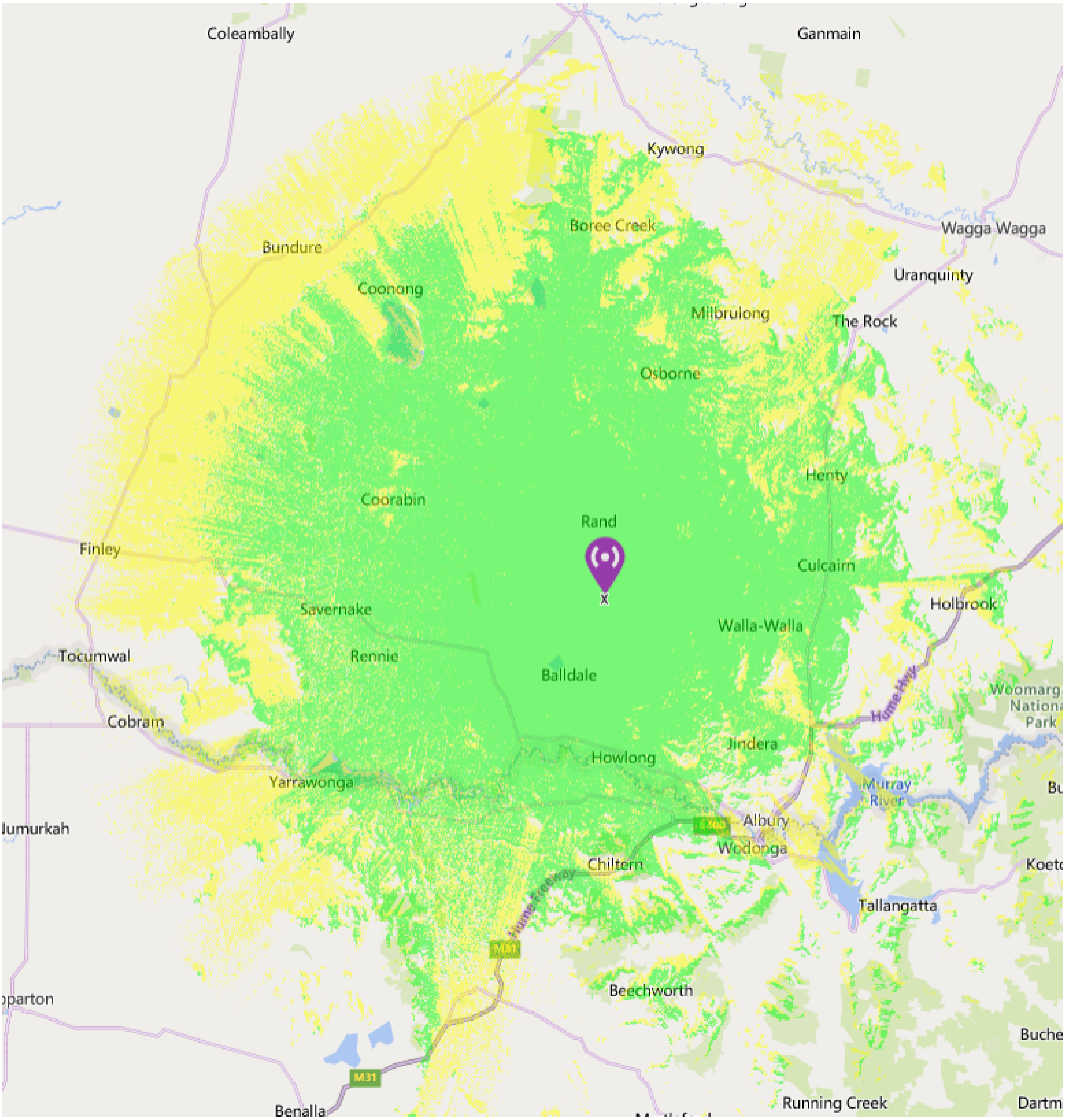
Matt suggested I connect on Wednesday night at 8.00pm for the weekly Net.
So I dialled in to 1302, VK2RWD.
The first thing you notice is the good quality audio.
None of the processed sound like EchoLink or IRLP has and certainly not the robotic sound of Dstar.
I guess being written for PABX use the audio is better, but it sounds better than telephone quality.
The first NEVARC 2 meter Net I ever joined had Frank VK2BFC as Net Control using VK3ANE, VK3KHS Shane, VK3FMW Steve, VK3VS Matt and VK3DTS Dave who was using a 5 watt handheld at Wangaratta which is over 60km distance as the RF goes to the VK3RWO repeater.



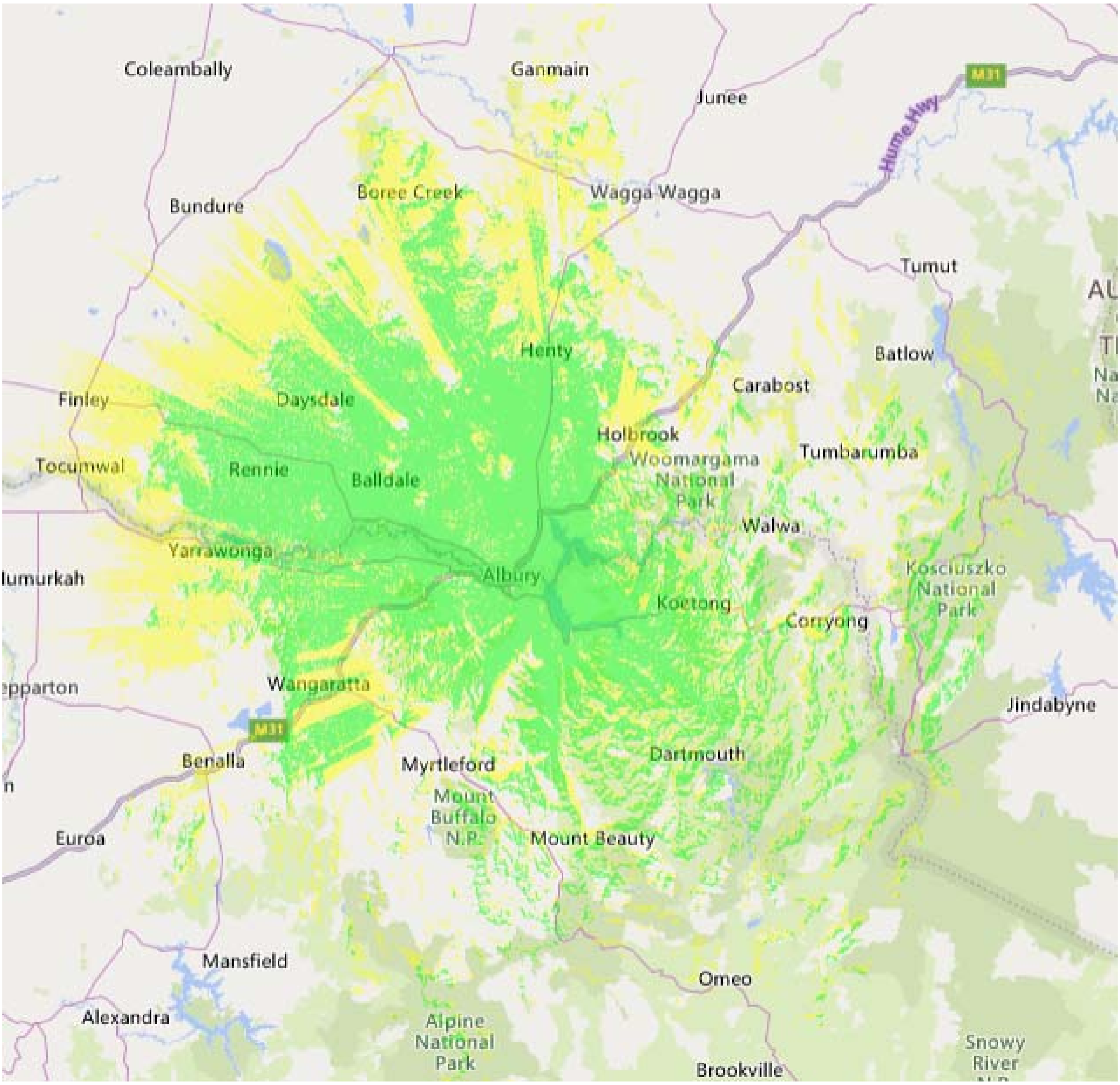
On the Net via the shack PC

On the Net via the kitchen laptop

Be on the Net anywhere with the mobile

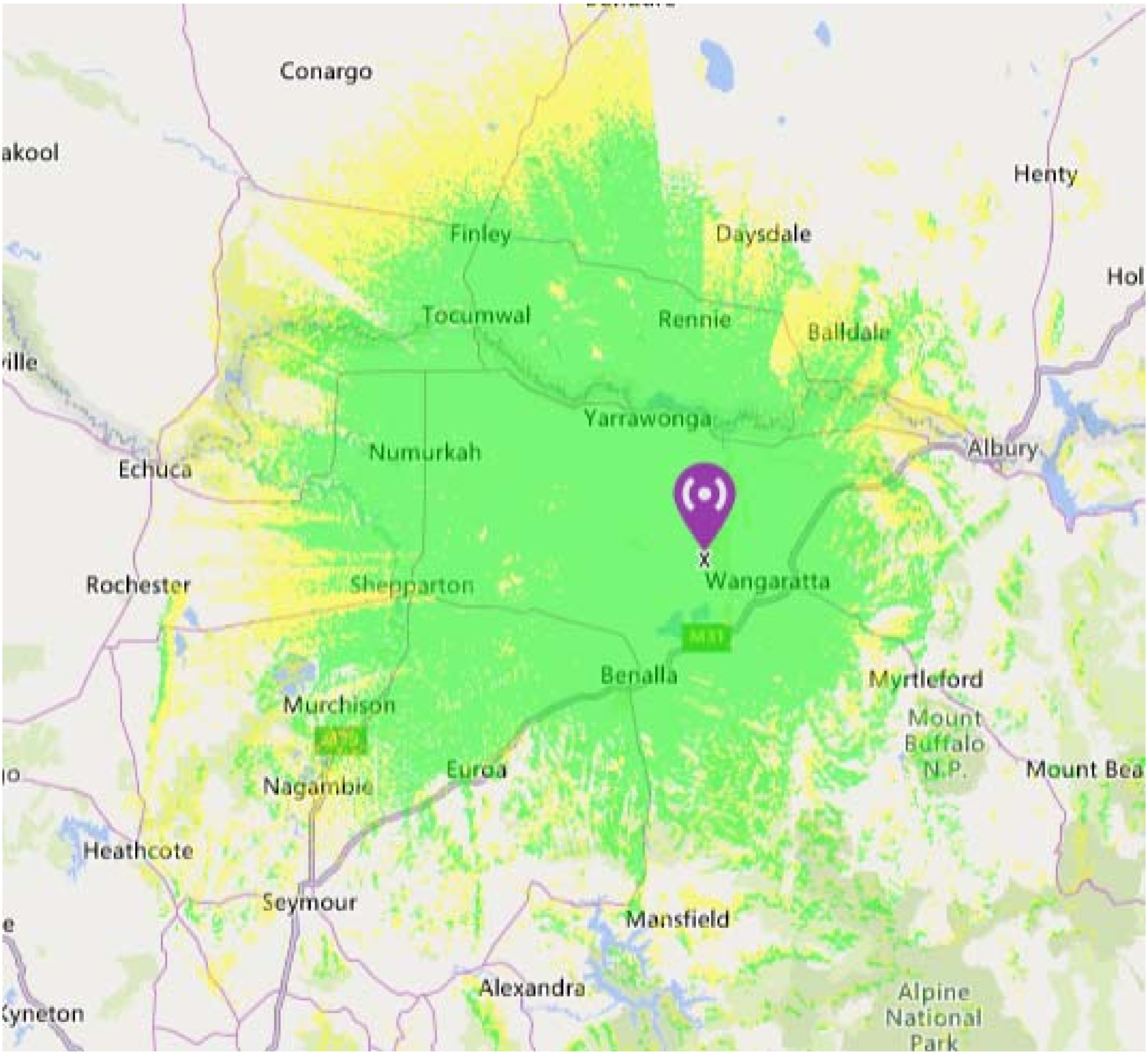


Propagation Map for VK2RWD 2 Meter Repeater



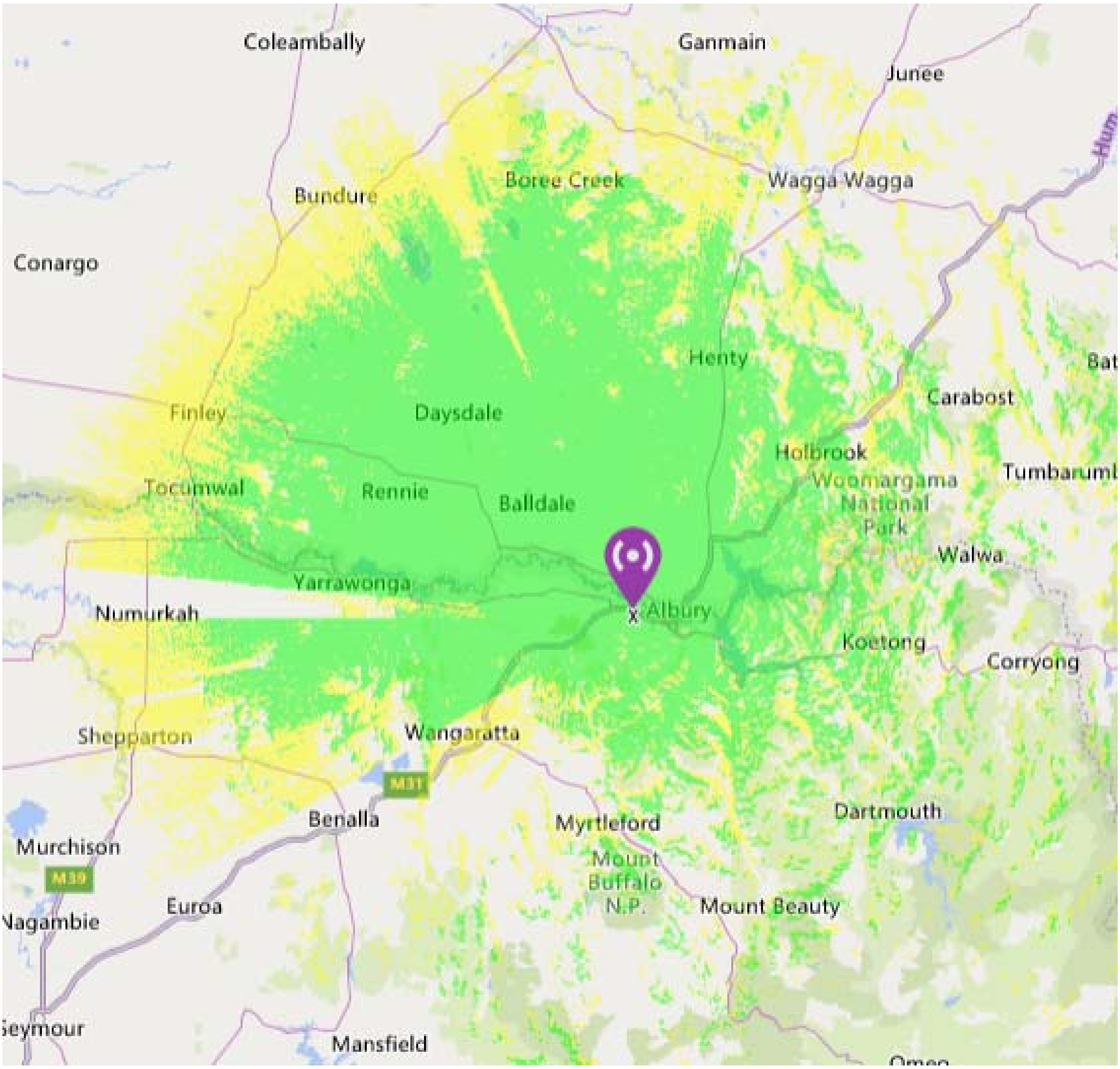
Propagation map for VK2RWO 2 Meter Repeater

Currently VK3RWO and VK3RWC are linked together via VKLink



Propagation map for VK3RWC 2 Meter Repeater

Currently VK3RWO and VK3RWC are linked together via VKLink



Propagation map for the proposed VK3RWO 2 Meter Repeater

Not yet in service

MATT DOES SOME ADJUSTMENTS

Matt has been doing some fiddling, shouldn't surprise any of you, for future 2 Meter nets, dial into the NET, via your IP phone by dialling 1301.
1301 is a pseudo node created that includes 1312 (VK3RWC), 1202 (VK2RWD), and 1302 (VK3RWO).

The reason Matt did this, is if any of the nodes end up on a small 4G connection, the nature of SIP, if its connected its sending data, albeit small bits, and we don't want that on a 4G connection. Whereas when you connect to the pseudo node, the data goes to it no worries, but the IAX data from the pseudo node to the rest of them only counts up when someone is transmitting, not all the time.
As it stands 1312 and 1202 are on NBN, and you can dial into them, even though you come out on all the connected devices, you can dial 1302 but it will reject you as it won't allow SIP connections.

Remember, * to TX and # RX

YOU NEED TO LEAVE A BREAK BETWEEN OVERS SO ROIP USERS CAN BREAK IN

The second NEVARC 2 meter Net I joined, after Matt made the adjustments, worked perfectly.

USING A VoIP TELEPHONE TO ACCESS NEVARC CLUB REPEATERS

Matt also suggested, due to the Optus modem not allowing to setup Telephony VoIP, a standalone VoIP Phone could be used. Matt said lots of cheap pre owned and refurbished VoIP Phones on EBay. Prices started at $15, with $35 or more getting a decent VoIP Phone, postage included.
An EBay search of "VoIP phone" brings up lots of phones, all in Australia.
I bought a Cisco Unified IP Phone 7931G 3-Line VOIP Phone & Stand for $29 posted. The features are complete overkill, but at $29 who cares.
This is a 2014 model, obsolete phone with no further service support or updates from Cisco since 2019, hence the cheap price, but will do for ham radio.
Later discovered it needs a 48 volt DC supply, so I had to order one of them, it can be powered by 'Power over Ethernet', but my Optus router doesn't support that feature. Or you use PoE [Power over Ethernet] adapters / injectors.

SETTING UP THE VoIP TELEPHONE

My VoIP phone has a user manual that can be downloaded at 260+ pages, being in COVID lockdown, I actually read it all. I had this idea that I could use the VoIP Phone for both landline and internet calls – but no. There is no provision to use your landline with these phones, but you can sign up to VoIP providers.
So for me, this phone is a dedicated "ham radio" device, no need to have your computer running to call.

Firstly after connecting the handset, Ethernet cable and power to the phone, its time to put in the software settings. The main job is to find the IP address of the phone and type that into a web browser.
It asks for a lot of optional settings, which can be ignored. Matt had previously emailed me all the necessary software settings for Zoiper5, so they were exactly the same settings to be programmed to the VoIP Phone. However the phone did not have SIP options on the menu. A factory reset was done and then the phone did not work at all, it kept rebooting in a loop.
Next try was to contact the EBay seller and see if they have any advice, I don't find much help on the internet as it is an unsupported model.

The EBay seller replied to me, but the info they gave me I had already found on the internet.

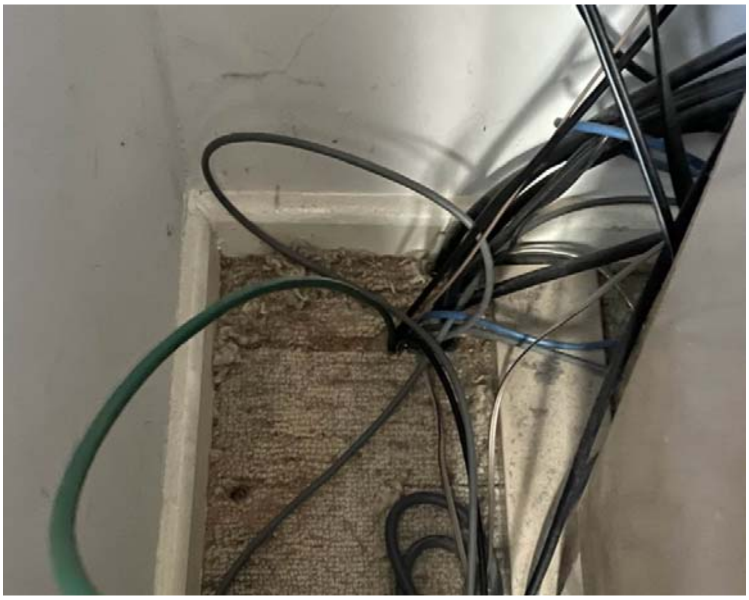
ANOTHER ETHERNET CABLE RUN UNDER THE HOUSE

The SIP phone will need an Ethernet cable from the router which is located in another room in the house.
As if there weren't enough cables already under the house, time to install another one.
I used to crawl under the house and get covered in dust until I found a better way.

Instead of crawling to the very tight narrow west side of the house underneath the radio room, I now use a spare X7000 fibreglass antenna to string the cable through to whatever area of the house required.
This is good for computer cables, DC power cables, telephone lines, audio cables and coax cables.

The job starts by poking the cable through the floor in the radio room, to identify the cable I got a green one.
Then its outside with the X7000 and pulling the green CAT8 cable outside on one end with a coat hanger.
The end of the CAT8 Green Ethernet cable is taped to the end of the X7000.
The X7000 is then fed with cable attached to the area of the house you want the cable run to go.
So with the right angle you can poke the X7000 to near the room you want.
Then I go underneath the house on the east side which is higher from the ground.
Next you take the tape off and recover the Ethernet cable and poke it up a hole where the router is located.

I could not remember what length the previous Ethernet cables were, so I guessed 10 meters instead of 15 meters. Later I discovered it was 15 meters usually, as the cable only had 5cm spare in the run.
Another 6 cm less and it would have been to go and buy a 15 meter length and crawl under the house again.



The green Ethernet [yet another cable] behind the desk

You have to take care with the Ethernet plugs so you don't snap the little clip on the plug.
I have always thought 'RJ' plugs are fickle, but you just have to take care when pulling them through.
Back inside the room where the router is located the Ethernet cable is labelled and plugged into the router.

Then it is back to the other side of the house and pull the X7000 back out and put it away.
This results in a dust free job for me and my clothes which keeps the XYL happy.

Back in the radio room it is time to also label the green Ethernet cable and decide where the SIP phone will go.
Then any spare cable can be pushed back under the house to keep things neat.

Remember – label and document all your cable runs, which do grow, over time, you will thank yourself later.

DIFFERENCES BETWEEN ETHERNET CATEGORIES

CAT5 used to be in use everywhere years ago, a few years ago I used CAT7 to run a remote head lead on an ICOM 7100 radio, as the dual shielding stopped the clicks going across to the microphone leads.

CAT5 operates at 100 MHz and can transfer data at speeds up to 1000 Mbps. CAT6 works at 250 MHz and can get up to 1 Gbps.
CAT7 ups the ante substantially with 600 MHz and 10 Gbps rates at 100 meters length.
CAT8 is a major upgrade as it is jumping several iterations in performance. It uses 2 GHz signals to move data from 25 Gbps (CAT8.1) to 40 Gbps (CAT8.2). CAT8 cable is rated for 2000MHz and 40GB at 30 meters.

Ethernet Speed Comparison

Cable Type	Shielding	Maximum Frequency	Potential Throughput
CAT 1	No	10 kHz	1 Mbps
CAT 2	No	1 MHz	4 Mbps
CAT 3	No	16 MHz	10 Mbps
CAT 4	No	16 MHz	10 Mbps
CAT 5	No	100 MHz	100 Mbps
CAT 5e	No	100 MHz	1 Gbps
CAT 6	Sometimes	250 MHz	1 Gbps
CAT 6a	Sometimes	500 MHz	10 Gbps
CAT 7	Yes	600 MHz	10 Gbps
CAT 8	Yes	2 GHz	40 Gbps

Higher frequencies require more twists in the cable pairs, and that process is more expensive.
Also, it becomes increasingly difficult to shield higher frequencies from interference and crosstalk, and as you go up in scale, the cost of raw materials for shielding get pricey.

CAT7, for example, often uses gold plates for shielding.
Since interference can render a cable useless, this is a big deal.

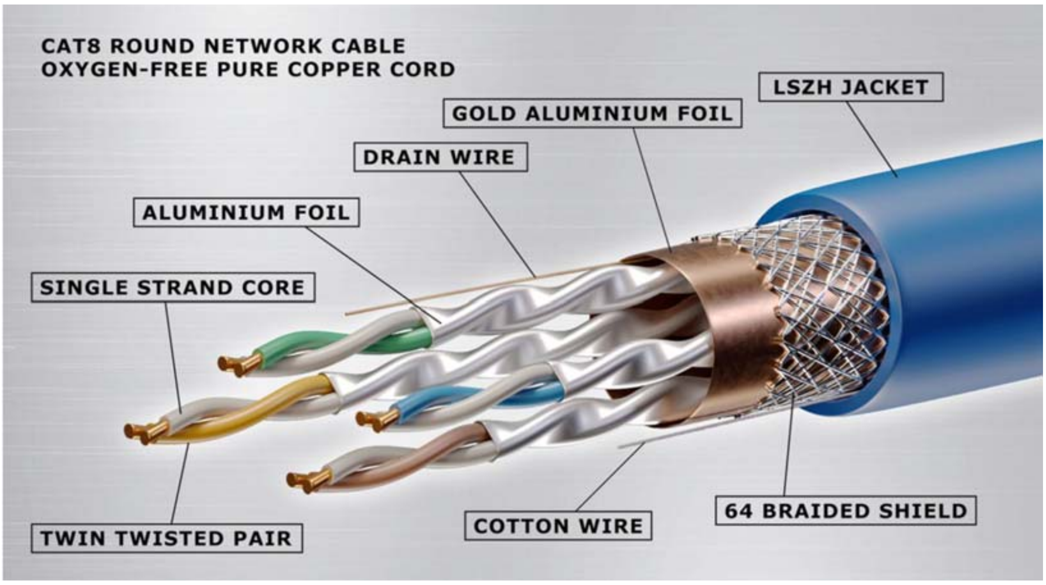
With CAT8, one of the popular and most trusted types of shielding is S/FTP.
This is usually established by having each pair shielded with a foil wrapping and then a 4-pair braid shield around the group of wires. This gives the maximum level of protection from interference and is found in the highest performance copper cables.

The frequency of a cable determines how many 1s and 0s can be sent across the wires in a second. For basic CAT5 cables, that's 100 million signals a second (or 100 MHz).

CAT8 uses an unprecedented 2 billion (2 GHz) signals per second. This means the cable density and quality of shielding necessary to make it work are on a whole different scale.
That's obvious when you consider that Cat8 is rated for data transfers 250 to 400 times faster than CAT5.

CAT8 can handle 10G, 25G and 40G networking demands at distances up to 30 meters.
It still works with RJ-45 connectors, but can also be sourced with Class II non-RJ45 connections when needed.

My choice of CAT8 Ethernet cable for the SIP Phone is total overkill, a bit more expensive, but as it was only a short 10 meter run, I got it. Maybe later I can swap the old Ethernet for the SIP Phone and run my internet from the router to the computer with the new CAT8 cable.

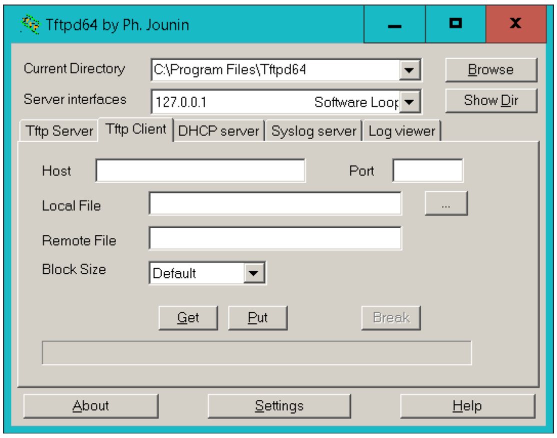


CONFIGURATION OF THE SIP PHONE

With the SIP Phone not having any SIP Manu options, I asked Matt VK3SMB for help.
Matt suggested that install a remote client on my PC and he remotely connected and had a play with the phone settings. After a lot of research Matt suggested I do a factory reset on the SIP phone.
I did a factory reset, but the SIP phone ended up in a reboot loop, probably because it is not in a business environment with no SIP phone server to pull its files from.

After more research I found the SIP phone needs an FTP Server with all its configuration files and settings to connect to. Looking at the Cisco website, I found [as they warned] that my model phone is no longer supported and to get files you required an account and pay for them.
Further looking elsewhere on the web, I found the exact model, 7931G files in a zipped package.
After scanning it for any nasties, I downloaded it and unzipped it and placed it in its own directory.
Hopefully it is an acceptable software version, that is compatible with the SIP phone I have.

Now I required an FTP Server.
After more research I found a suitable program called TFTPD32 which was installed on my PC.



Next it was to point the directory to where I unzipped the SIP phone files, then to set up DHCP settings.
For DHCP settings I was unsure if TFTPD32 could do the job, or if I had to use my Optus router for this.
Matt had doubts my Optus router would cooperate but to have a go anyway.

PERFORMING A FACTORY RESET – FROM THE CISCO 7931G ADMINISTRATION MANUAL

When you perform a factory reset of the Cisco Unified IP Phone, the following information is erased or reset to its default value:

• CTL file—Erased
• LSC—Erased
• User configuration settings—Reset to default values
• Network configuration settings—Reset to default values
• Call histories—Erased
• Locale information—Reset to default values
• Phone application—Erased
(phone recovers by loading the term70.default.loads file or the term71.default.loads file, depending on the phone model)

Before you perform a factory reset, ensure that the following conditions are met:

• The phone must be on a DHCP-enabled network.
• A valid TFTP server must be set in DHCP option 150 or option 66 on the DHCP server.
• The term70.default.loads file or the term71.default.loads file and the files specified in that file should be available on the TFTP server that is specified by the DHCP packet.

To perform a factory reset of a phone, perform the following steps:

Step 1

Unplug the power cable from the phone and then plug it back in.
The phone begins its power up cycle.

Step 2

While the phone is powering up, and before the Speaker button flashes on and off, press and hold #.
Continue to hold # until each line button flashes on and off in sequence in amber.

Step 3

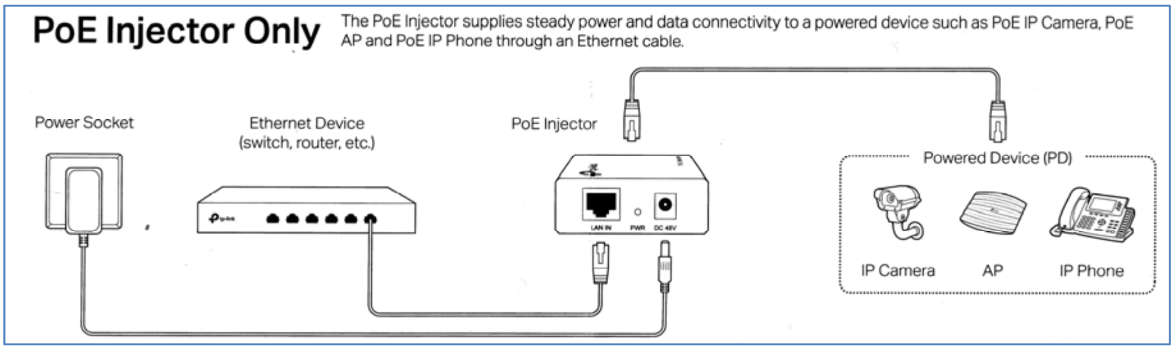
Release # and press 123456789*0#.
You can press a key twice in a row, but if you press the keys out of sequence, the factory reset will not take place. After you press these keys, the line buttons on the phone flash amber and then green, and the phone goes through the factory reset process. This process can take several minutes.
Do not power down the phone until it completes the factory reset process and the main screen appears.

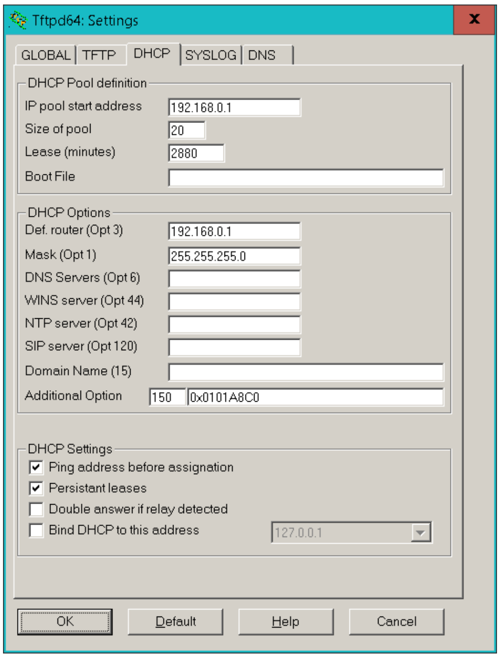
When I did a factory reset prior, I did not have the FTP server and files for the SIP phone to find, hence the rebooting loop as it could not find what it was looking for.

So after setting everything I needed up, it was time, with some stress, to have another try and see if I can "unbrick" the SIP phone.

CONFIGURATION OF THE SIP PHONE CONTINUED

Powering the SIP Phone is done using a PoE injector, simple and cheap.



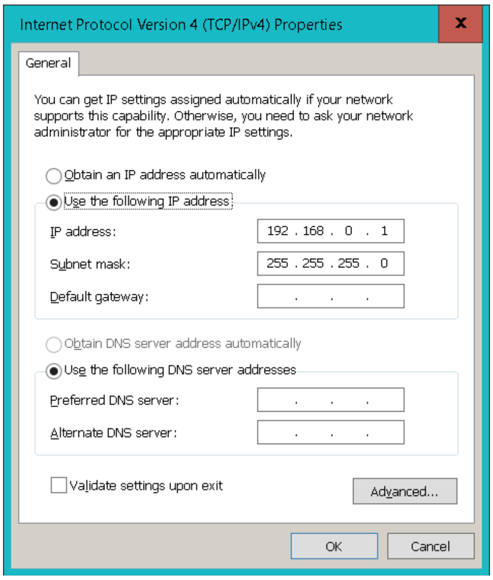


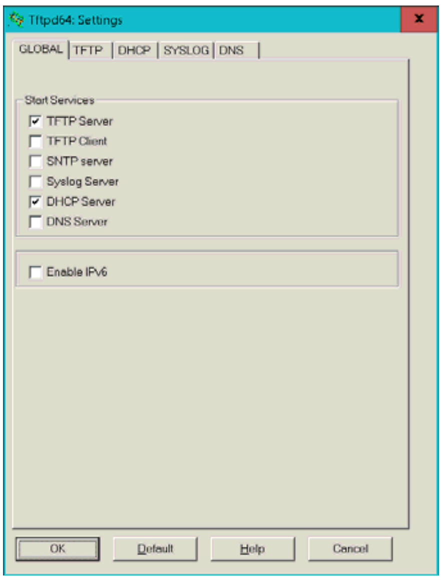
After a few days of IP conflicts Matt said, "The router is giving it 0.6 based on its MAC address.
The TFTP server is fighting with your router. Let TFTP server do the DHCP if it can, failing that, install a DHCP server, add the TFTP server with option 150 to it and remove your router for the minute.

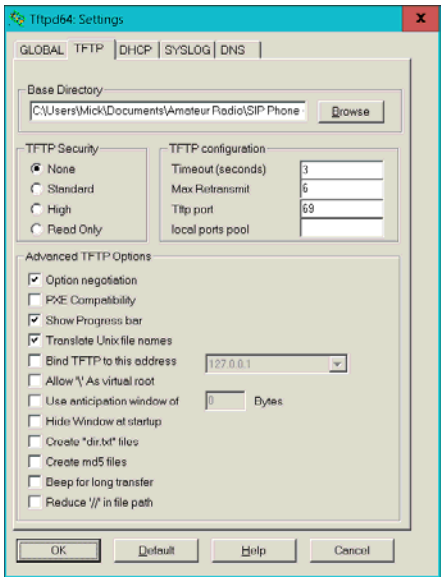
You are going to need a switch between PC and phone if your router is missing."

So after another visit to get a router, it was time for another try.
The router was used for data communication between the PC and SIP Phone, with the Optus modem removed.

TFTPD32 was used with DHCP option 150 selected. I gave the PC a static IP of 192.168.0.1 with router disconnected, with only PC & SIP Phone in switch, the Optus router was disconnected.







I use [after powering phone with # pressed] 123456789*0# this reboots the SIP Phone and has it look for an IP address and then download a flash file.
I also tried the hard reboot of 3491672850*# but this was no good either.
I am going to order another SIP Phone and try again with that, so no results until next month.

A BRIEF HISTORY OF THE ASTERISK PROJECT

Linux Support Services

Way, way back in 1999 a young man named Mark Spencer was finishing his Computer Engineering degree at Auburn University when he hit on an interesting business concept. 1999 was the high point in the .com revolution (aka bubble), and thousands of businesses world-wide were discovering that they could save money by using the open source Linux operating system in place of proprietary operating systems. The lure of a free operating system with open access to the source code was too much to pass up. Unfortunately there was little in the way of commercial support available for Linux at that time. Mark decided to fill this gap by creating a company called "Linux Support Services". LSS offered a support hotline that IT professionals could (for a fee) call to get help with Linux.

The idea took off. Within a few months, Mark had a small office staffed with Linux experts. Within a few more months the growth of the business expanded demanded a "real" phone system that could distribute calls evenly across the support team, so Mark called up several local phone system vendors and asked for quotes. Much to his surprise, the responses all came back well above $50,000 -- far more than Mark had budgeted for the project. Far more than LSS could afford.

Finding a Solution

Rather than give in and take out a small business loan, Mark made a pivotal decision. He decided to write his own phone system. Why not? A phone system is really just a computer running phone software, right? Fortunately for us, Mark had no idea how big a project he had take on. If he had known what a massive undertaking it was to build a phone system from the ground up might have gritted his teeth, borrowed the money and spent the next decade doing Linux support. But he didn't know what he didn't know, and so he started to code. And he coded. And he coded.

Mark had done his engineering co-op at Adtran, a communications and networking device manufacturer in Huntsville, AL. There he had cut his teeth on telecommunications system development, solving difficult problems generating a prodigious amount of complex code in short time. This experience proved invaluable as he began to frame out the system which grew into Asterisk. In only a few months Mark crafted the original Asterisk core code. As soon as he had a working prototype he published the source code on the Internet, making it available under the GPL license (the same license used for Linux).

Within a few months the idea of an "open source PBX" caught on. There had been a few other open source communications projects, but none had captured the imagination of the global population of communications geeks like Asterisk. As Mark laboured on the core system, hundreds (now thousands) of developers from all over the world began to submit new features and functions.

Digium

What became of Linux Support Services? In 2001, Linux Support Services changed its name to Digium. Digium continued to develop Asterisk in collaboration with the community, provide services to support the development community, as well as build commercial products and services around Asterisk which have fuelled growth in both Digium and the Asterisk project. You can find out more about Digium at the Sangoma website and on wikipedia.

Asterisk in the Present

Asterisk is constantly evolving to meet the needs of the project's user-base. It's difficult to summarize the vast scope of everything that Asterisk can do as a communications toolkit.

The History of App_Rpt

By Jim Dixon WB6NIL

Once upon a time, there was Asterisk PBX. I was heavily involved in its initial practical implementation if interested; see below *(Zapata Telephony Project as it relates to the Asterisk PBX*).

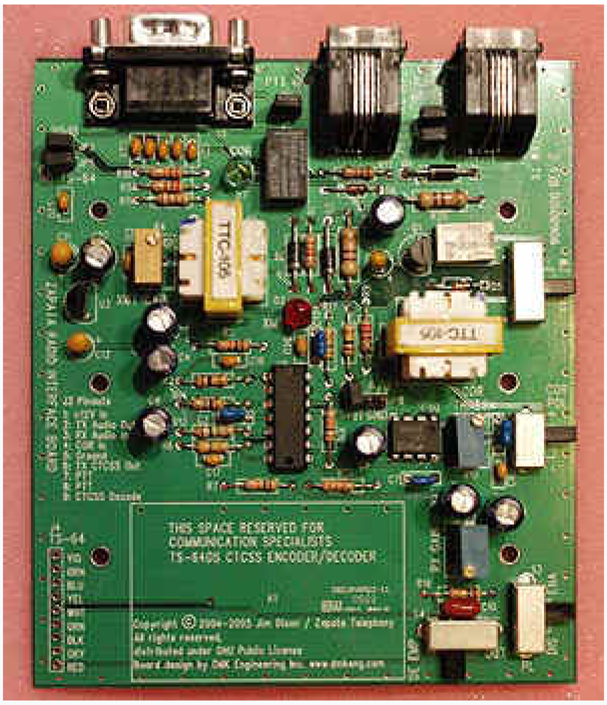
From its very onset, I saw that Asterisk was not only a good telephony switch, but also makes a good application implementation platform for virtually any telecommunication application that requires use of many of the things that Asterisk provides.

App_Rpt was started, not only to demonstrate this feature of Asterisk, but to essentially provide an outlet of nearly 30 years of experience and frustration with repeater and remote base systems, in the form of designing a system essentially "the way I think a system should be."

I learned from and was inspired by some of the best ideas and implementations and also from some of the worst, and tried to create a workable and desirable system that met as many needs as possible.

My first big hurdle was to design a radio interface that could interconnect with a PC internally and the typical interface signals of a two-way radio. In the interest of keeping the technology simple and inexpensive, I designed a board (the ARIB board) which interfaced 2 analog FXS ports (either on a multiport analog card, like the Digium TDM400P, or a channel bank connected to the PC through a T1 card, that is already part of an existing phone switch) to the Radio signals (and provide buffering, level control, etc).

Analog Radio Interface



Rev. C. Analog Radio Interface Board

This interface *did* work well to allow me to do the initial development of the software, and first half a dozen deployments, since most of us that had it first already had Asterisk phone switches already in operation, with spare ports on our channel banks.

The systems were originally deployed in Northern California at a couple of locations, and in San Diego by Steve, WA6ZFT.

With Steve's greatly appreciated help, we were able to debug the thing and add many new and interesting features.

PCI Radio Card

Quad PCI Radio Card

After a few months of the system running stably (despite new additions all the time), we decided that better system integration was appropriate, so we (along with help from David Kramer) designed and produced the Quad PCI Radio Interface card. This allows interface between the PC and 4 separate radio systems simultaneously, via 8 pin modular connectors.

In addition we also found an optimal PC in which to best use these PCI cards. It’s a mini-ITX system that runs on 12 volts directly, 1 U rackmount and has no moving parts, and is comparatively inexpensive.

Together the PCI card and the PC make an ultra-stable, super high-end radio controller system that, coupled with its open-source-ness gives the system implementer supreme flexibility and functionality.

For the next 2 years, we kept adding features, and refining the ones we had, and discovering that even though MANY people were QUITE interested in the technology, the price for entry was a bit high (for those who had no experience and wanted to just evaluate it). It became painfully obvious that something had to be done, somehow, to allow the hardware interface price to be lowered considerably without sacrifice in the quality of the technology.

USB

DMK Engineering URI

After looking at many possibilities, including usage of a PC's internal audio subsystem (yuch!), it sure seemed that the only realistic chance we had was to get something reliably working with a USB interface.

Experiments with USB several years previously show it to be utterly flaky and useless on Linux, and not much better even on Windows. But, since that's about all that's available any more (other then maybe Firewire, which is more or less the same thing) I figured we should re-visit the possibility of stable USB, and did so and found success. It’s good that things change.

After some looking around, I found the C-Media CM-108 USB Sound IC. It is VERY simple, VERY high quality, VERY reliable and VERY inexpensive. Even already make USB sound cards (well, sound devices, they're awfully small to call a card) that could easily be modified for development and testing purposes.

So after some testing and verification of the USB hardware, I went to my friend Steve Henke, W9SH (who had already done some major work on the IAXRPT project for us), and asked him to use his amazing DSP skills to write code to do two-way audio processing right in the PC (going along with the Zapata Telephony motto of *"We don't need no stinkin' DSP!"*), which, happily he did.

We now have functional DSP code and channel interface code to allow USB devices to interface with App_Rpt and Asterisk, allowing high-quality, yet inexpensive interconnection with radio devices.

Zapata Telephony and how it relates to Asterisk PBX

By Jim Dixon, WB6NIL

About 20-25 or so years ago, AT&T started offering an API (well, one to an extent, at least) allowing users customize functionality of their Audix voicemail/attendant system which ran on an AT&T 3BX usually 3B10) Unix platform. This system cost thousands of dollars a port, and had very limited functionality.

In an attempt to make things more possible and attractive (especially to those who didn’t have an AT&T PBX or Central Office switch to hook Audix up to) a couple of manufacturers came out with a card that you could put in your PC, which ran under MS-DOS, and answered one single POTS line (loopstart FXO only). These were rather low quality, compared with today's standards (not to mention the horrendously pessimal environment in which they had to run), and still cost upwards of $1,000 each. Most of these cards ended up being really bad sounding and flaky personal answering machines.

In 1985 or so, a couple of companies came out with pretty-much decent 4 port cards, that cost about $1,000 each (wow, brought the cost down to $250 per port!). They worked MUCH more reliably then their single port predecessors, and actually sounded pretty decent, and you could actually put 6 or 8 of them in a fast 286 machine, so a 32 port system was easy to attain. As a result the age of practical Computer Telephony had begun.

As a consultant, I have been working heavily in the area of Computer Telephony ever since it existed. I very quickly became extremely well- versed in the hardware, software and system design aspects of it. This was not difficult, since I already had years of experience in non-computer based telephony.

After seeing my customers (who deployed the systems that I designed, in VERY big ways) spending literally millions of dollars every year (just one of my customers alone would spend over $1M/year alone, not to mention several others that came close) on high density Computer Telecom hardware.

It really tore me apart to see these people spending $5,000 or $10,000 for a board that cost some manufacturer a few hundred dollars to make. And furthermore, the software and drivers would never

work 100% properly. I think one of the many reasons that I got a lot of work in this area, was that I knew all the ways in which the stuff was broken, and knew how to work around it (or not).

In any case, the cards had to be at least somewhat expensive, because they had to contain a reasonable amount of processing power (and not just conventional processing, DSP functionality was necessary), because the PC's to which they were attached just didn’t have much processing power at that time.

Very early on, I knew that someday in some "perfect" future out there over the horizon, it would be commonplace for computers to handle all of the necessary processing functionality internally, making the necessary external hardware to connect up to telecom interfaces VERY inexpensive and in some cases trivial.

Accordingly, I always sort of kept a corner of an eye out for what the "Put on your seatbelts, you’ve never seen one this fast before" processor throughput was becoming over time, and in about the 486-66DX2 era, it looked like things were pretty much progressing at a sort of fixed exponential rate. I knew, especially after the Pentium processors came out, that the time for internalization of Computer Telephony was going to be soon, so I kept a much more watchful eye out.

I figured that if I was looking for this out there, there **must** be others thinking the same thing, and doing something about it. I looked, and searched and waited, and along about the time of the PentiumIII-1000 (100 MHz Bus) I finally said, "gosh these processors CLEARLY have to be able to handle this".

But to my dismay, no one had done anything about this. What I hadn't realized was that my vision was 100% right on, I just didn’t know that **I** was going to be one that implemented it.

In order to prove my initial concept I dug out an old Mitel MB89000C "ISDN Express Development" card (an ISA card that had more or less one-of-everything telecom on it for the purpose of designing with their telecom hardware) which contained a couple of T-1 interfaces and a cross-point matrix (Timeslot- Interchanger). This would give me physical access from the PC's ISA bus to the data on the T-1 timeslots (albeit not efficiently, as it was in 8 bit I/O and the TSI chip required MUCHO wait states for access).

I wrote a driver for the kludge card (I had to make a couple of mods to it) for FreeBSD (which was my OS of choice at the time), and determined that I could actually reliably get 6 channels of I/O from the card. But, more importantly, the 6 channels of user-space processing (buffer movement, DTMF decoding, etc), barely took any CPU time at all, thoroughly proving that the 600MHZ PIII I had at the time could probably process 50-75 ports if the BUS I/O didn’t take too much of it.

As a result of the success (the 'mie' driver as I called it) I went out and got stuff to wire wrap a new ISA card design that made efficient use of (as it turns out all of) the ISA bus in 16 bit mode with no wait states. I was successful in getting 2 entire T-1's (48 channels) of data transferred over the bus, and the PC was able to handle it without any problems.

So I had ISA cards made, and offered them for sale (I sold about 50 of them) and put the full design (including board photo plot files) on the Net for public consumption.

Since this concept was so revolutionary, and was certain to make a lot of waves in the industry, I decided on the Mexican revolutionary motif, and named the technology and organization after the famous Mexican revolutionary Emiliano Zapata. I decided to call the card the "tormenta" which, in Spanish, means "storm", but contextually is usually used to imply a " **BIG** storm", like a hurricane or such.

That's how Zapata Telephony started.

I wrote a complete driver for the Tormenta ISA card for *BSD, and put it out on the Net. The response I got, with little exception was "well that's great for BSD, but what do you have for Linux?"

Personally, Id never even seen Linux run before. But, I can take a hint, so I went down to the local store (Fry's in Woodland Hills) and bought a copy of RedHat Linux 6.0 off the shelf (I think 7.0 had

JUST been released but was not available on shelf yet). I loaded it into a PC, (including full development stuff including Kernel sources). I poked around in the driver sources until I found a VERY simple driver that had all the basics, entry points, interfaces, etc (I used the Video Spigot driver for the most part), and used it to show me how to format (well at least to be functional) a minimal Linux driver. So, I ported the BSD driver over to Linux (actually wasn't **that** difficult, since most of the general concepts are roughly the same). It didn't have support for loadable kernel modules (heck what was that? in BSD 3.X you have to re-compile the Kernel to change configurations. The last system I used with loadable drivers was VAX/VMS.) but it did function (after you re-compiled a kernel with it included). Since my whole entire experience with Linux consisted of installation and writing a kernel module, I **knew** that it **had** to be just wrong, wrong, wrong, full of bad, obnoxious, things, faux pauses, and things that would curl even a happy Penguin's nose hairs.

With this in mind, I announced/released it on the Net, with the full knowledge that some Linux Kernel dude would come along, laugh, then barf, then laugh again, then take pity on me and offer to re-format it into "proper Linuxness".

Within 48 hours of its posting I got an email from some dude in Alabama (Mark Spencer), who offered to do exactly that. Not only that he said that he had something that would be perfect for this whole thing (Asterisk).

At the time, Asterisk was a functional concept, but had no real way of becoming a practical useful thing, since it didn't, at that time, have a concept of being able to talk directly (or very well indirectly for that matter, being that there wasn't much, if any, in the way of practical VOIP hardware available) to any Telecom hardware (phones, lines, etc). Its marriage with the Zapata Telephony system concept and hardware/driver/ library design and interface allowed it to grow to be a real switch that could talk to real telephones, lines, etc.

Additionally Mark has nothing short of brilliant insight into VOIP, networking, system internals, etc., and at the beginning of all this had a great interest in Telephones and Telephony. But he had limited experience in Telephone systems, and how they work, particularly in the area of telecom hardware interfaces. From the beginning I was and always have been there, to help him in these areas, both providing information, and implementing code in both the drivers and the switch for various things related to this. We, and now more recently others have made a good team (heck I ask him stuff about kernels, VOIP, and other really esoteric Linux stuff all the time), working for the common goal of bringing the ultimate in Telecom technology to the public at a realistic and affordable price.

Since the ISA card, I designed the "Tormenta 2 PCI Quad T1/E1" card, which Mark marketed as the Digium T400P and E400P, and now Varion is marketing as the V400P (both T1 and E1). All of the design files (including photo plot files) are available on the Zapatatelephony.org website for public consumption.

We have more, higher-density designs on the way.

As anyone can see, with Mark's dedicated work (and a lot of Mine and other people's) on the Zaptel drivers and the Asterisk software, the technologies have come a long, long way, and continue to grow and improve every day.

Footnote:

Has anyone ever taken a moment to sit back and consider the ENORMOUS responsibility that Mark has taken upon himself by doing this project? Have you ever thought of how incredibly many things that he has to concern himself with, and that it just **NEVER ENDS**! At this point, I believe that I have worked with him on this project longer that just about anyone, including some of his employees, and believe me, I have a good vantage point to see at least some of the stuff that he has to go through to accomplish this.

Personally, I would have **NEVER** taken on such a task, being that I am and was quite aware of the level of responsibility required to do so.

Yes, the task that I took on was and is quite a task, and quite a responsibility, but I did what I knew I could accomplish. Mark's part is way larger then mine, and all I can say that I know what it takes for him to do what he is doing, and I seriously appreciate the time and dedication that he has put into all the incredibly wonderful things that he has done for it and all of us.

Furthermore, Id like to seriously thank all of the project contributors and everyone else that has done some part to help with this project. Thank you for demonstrating that you believe in it, and that you believe in us.

Jim Dixon, WB6NIL Editors note: Jim became a SK in December of 2016. My best guess it these articles were written around 2006. I've applied minor formatting changes for readability.

Asterisk is a software implementation of a private branch exchange (PBX).

In conjunction with suitable telephony hardware interfaces and network applications, Asterisk is used to establish and control telephone calls between telecommunication endpoints, such as customary telephone sets, destinations on the public switched telephone network (PSTN), and devices or services on voice over Internet Protocol (VoIP) networks. Its name comes from the asterisk (*) symbol for a signal used in dual-tone multi-frequency (DTMF) dialling.

Asterisk was created in 1999 by Mark Spencer of Digium, which since 2018 is a division of Sangoma Technologies Corporation.
Originally designed for Linux, Asterisk runs on a variety of operating systems, including NetBSD, OpenBSD, FreeBSD, macOS, and Solaris, and can be installed in embedded systems based on OpenWrt.

The Asterisk software includes many features available in commercial and proprietary PBX systems: voice mail, conference calling, interactive voice response (phone menus), and automatic call distribution. Users can create new functionality by writing dial plan scripts in several of Asterisk's own extensions languages, by adding custom loadable modules written in PHP or C, or by implementing Asterisk Gateway Interface (AGI) programs using any programming language capable of communicating via the standard streams system (stdin and stdout) or by network TCP sockets.

Asterisk supports several standard voice over IP protocols, including the Session Initiation Protocol (SIP), the Media Gateway Control Protocol (MGCP), and H.323. Asterisk supports most SIP telephones, acting both as registrar and back-to-back user agent. It can serve as a gateway between IP phones and the PSTN via T- or E-carrier interfaces or analog FXO cards. The Inter-Asterisk eXchange (IAX) protocol, RFC 5456, native to Asterisk, provides efficient trunking of calls between Asterisk PBX systems, in addition to distributing some configuration logic. Many VoIP service providers support it for call completion into the PSTN, often because they themselves have deployed Asterisk or offer it as a hosted application. Some telephones also support the IAX protocol.

By supporting a variety of traditional and VoIP telephony services, Asterisk allows deployers to build telephone systems, or migrate existing systems to new technologies. Some sites are using Asterisk to replace proprietary PBXes, others provide additional features, such as voice mail or voice response menus, or virtual call shops, or to reduce cost by carrying both local and long-distance calls over the Internet.

In addition to VoIP protocols, Asterisk supports traditional circuit-switching protocols such as ISDN and SS7. This requires appropriate hardware interface cards, marketed by third-party vendors. Each protocol requires the installation of software modules. In Asterisk release 14 the Opus audio codec is supported.

~ Jim Dixon, WB6NIL [SK]

NOTES FROM MATT VK3VS

Its not long after this article that app_rpt started being forked...

Jim kept his code going, and released it in the "ACID" centos installation (the first one I used).

Then about the same time around 2009, I started making what is now VKLink, and Hamvoip started their fork.

At first we were swapping bits of code, then they slammed the door shut, so I went on my merry way.

The original repository of the code was closed not long after Jim's death.

I cloned the repository just before that happened.

The main differences between VKLink and Allstar and Hamvoip, is I can control the GPIO's of the raspberry Pi directly in my code, meaning I don't need to modify fobs etc to make it happen.

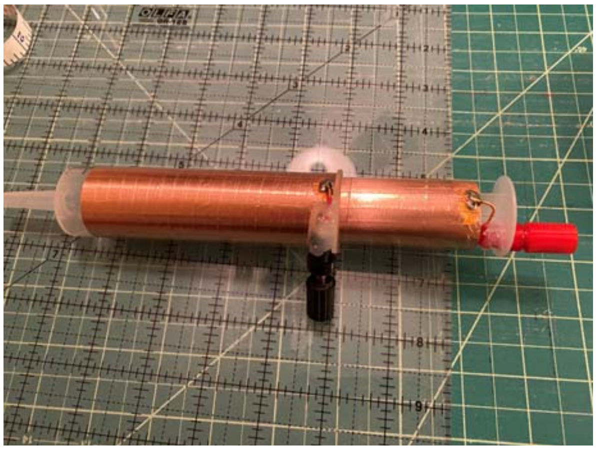
Allstar and Hamviop still rely on specialist devices or modified fobs.

Just to be clear, there is a major storm over licensing of app_rpt since Jim Dixon passed away.
Hamvoip have closed the code.

Whilst my VKLink code is not published publicly on the internet, I give it to whoever asks for it.

~Matt VK3VS

USES FOR A SPARE COVID NEEDLE



Amateur Radio as a hobby has a long history of encouraging experimentation using whatever one might have on hand. When Tom wanted to use his 14 MHz antenna outside of its designed frequency range, he knew he'd need an impedance matching circuit.

The most common type is an L-Match circuit which uses a variable capacitor and a variable inductor to adjust the usable frequency range (resonance) of an antenna. While inefficient in some specific configurations, they excel at bridging the gap between the 50 ohm impedance of the radio and the unknown impedance of an antenna.

No doubt raiding his junk box for parts, [Tom] hacked together a variable capacitor and inductor using ferrite rods from AM radios, hot glue, magnet wire, copper tape, and some surplus 60ml syringes. You can see that he ground out the centre of the plunger to make room for ferrite rods. Winding the outside of the syringe with magnet wire, the alignment of the ferrite can be adjusted via the plunger, changing the characteristics of the element to tune the circuit. Tom reports that he was able to make an on-air contact using his newly made tuner, and we're sure he enjoyed putting his improvised equipment to use.

DESCRIPTION
Sharing this variable inductor I threw together last weekend. 22AWG magnet wire.
Taps at 6,6,12,12,~90+ turns; 3 ferrite rods from some junk bin AM radios glued into the shaft of a 60ml syringe.
Using a 1/4 wave vertical wire cut for the middle of 20m and a moderate radial field this allowed me to impedance match anything from 160 to 30.
Efficiency is probably shoddy at 160, but I made a QSO on 40.

Also made a varicap on the same base.
In this config @ 14 MHz it gives 26-200 pF.

~From Hackaday, Internet

List of repeaters linked into the 9th 2021 DATV QSO Party

Southern California Chapter (ATN and ATN-CA) **Roland [KC6JPG]** - Control for California, Nevada, and Arizona

Jobs Peak - W6ATN
Mount Wilson - W6ATN
Oat Mountain - W6ATN
Ord Mountain - W6ATN
Santa Barbara - WB9KMO
Santiago Peak - W6ATN
Snow Peak - W6ATN

Arizona Chapter (ATN-AZ) **Jim [K6SOE]**

W7ATN - White Tank Mountain (Phoenix, AZ)
W7ATN - Mt. Lemmon (Tucson, AZ)
W7ATN - Greens Peak (North East AZ)

Florida Chapter (ATN-FL) - Wolfgang (KV4ATV)

KA4ATV - Panama City, FL

Nevada Chapter (ATN-NV) Las Vegas N7ZEV

Las Vegas - N7ZEV

Boulder ATV Club (ATN-CO) - BATVC **Jim [KH6KTV]**

KH6KTV - ?Boulder,CO

Amateur Television in Central Ohio (ATCO) **Art [WA8RMC]**

?WA8RMC - Columbus Ohio

Sydney ATV Group - **Gary [VK2CRJ]**
VK2RTS - Lawson Blue Mtns

Whyalla Amateur Radio Club - **Bevan [VK5BD]**

K5RDC - The Bluff Port Pirie & Whyalla

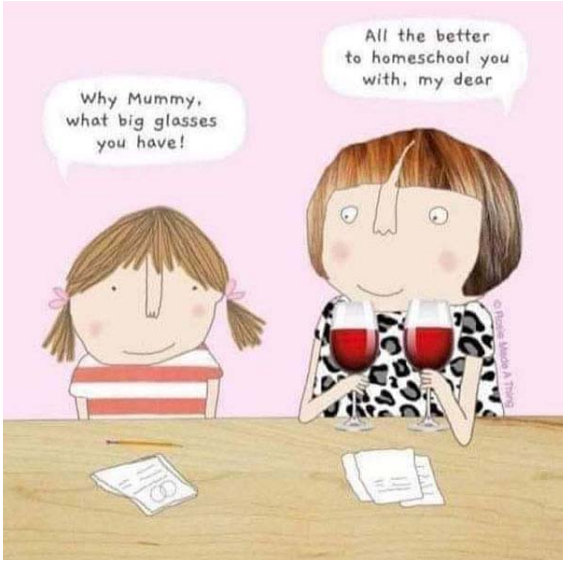
Melbourne Amateur TV Group - **Peter [VK3BFG]**

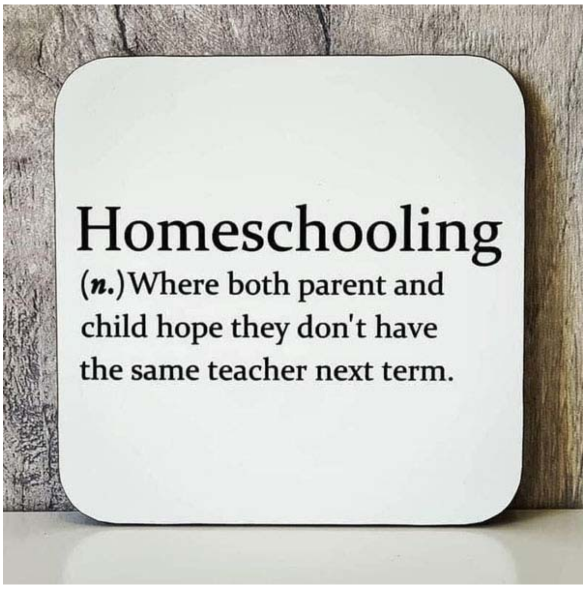
VK3RTV - Mount View, Melbourne

List of stations that participated in the 9th 2021 ATV QSO Party

(In callsign sort order)
01 Chris K0CJG
02 Ed K0JOY
03 Jim K6SOE
04 David KB4ICU
05 Wolfgang KC6AV
06 Roland KC6JPG
07 Don KE6BXT
08 Jim KH6HTV
09 Don KN6TRZ
10 Don N0YE
11 Bob N6AZB
12 Keith N6GKB
13 Frank N7ZEV
14 John VK2ATV
15 Gary VK2CRJ
16 Paul VK2JPL
17 John VK3ATV
18 Neil VK3BCU
19 Peter VK3BFG
20 Mick VK3CH
21 Clint VK3CSJ
22 Geoff VK3GE
23 Phil VK3GMZ
24 Ian VK3QL
25 Richard VK3VRS
26 Dennis VK3WV
27 Bevan VK5BD
28 Dave VK5DMC
29 David VK5DMC
30 John VK5KJG
31 Roger VK5YYY
32 Gary W6KVC
33 Tom W6ORG
34 Art WA8RMC
35 Fred WB6ASU
36 Tom WB6HYH
37 Dale WB8CJW
38 Dale WB8KPQ
39 Rod WB9KMO


YOU THINK IT'S BAD NOW?...
IN 20 YEARS THE COUNTRY'S GOING TO BE RUN BY KIDS THAT WERE HOMESCHOOLED BY ALCOHOLICS


Why Mummy, what big glasses you have!
All the better to homeschool you with, my dear


Homeschooling
(n.) Where both parent and child hope they don't have the same teacher next term.

NEVARC Net

40 Meter Net

7 Days a Week
10am Local time
(East coast)

7.097 MHz LSB

Approximately + or – QRM

Hosted by Ron VK3AHR

"Australia Ham Radio 40 Meter Net"

President, VK3VS, Matt
Vice President, VK2VU, Gary
Secretary, VK2BFC, Frank
Treasurer, Amy Bilston

WIA Affiliated Club

The Radio Amateur Society of Australia

NEVARC CLUB PROFILE

NEVARC
North East Victoria Amateur Radio Club

History

The North East Victoria Amateur Radio Club (NEVARC) formed in 2014.
As of the 7th August 2014, Incorporated, Registered Incorporation number A0061589C.
NEVARC is an affiliated club of the Wireless Institute of Australia and The Radio Amateur Society of Australia Inc.

Meetings

Meetings details are on the club website, the Second Sunday of every month, check for latest scheduled details.
Meetings held at the Belviour Guides Hall, 6 Silva Drive West Wodonga.
Meetings commence with a BBQ (with a donation tin for meat) at 12pm with meeting afterwards.
Members are encouraged to turn up a little earlier for clubroom maintenance.
Call in Via VK3RWO, 146.975, 123 Hz tone.

NEVARC NETS

HF
7.097 MHz 7 Days a Week - 10am Local time

VHF
VK2RWD Wednesday - 8.00pm Local time

Benefits

To provide the opportunity for Amateur Radio Operators and Short Wave Listeners to enhance their hobby through interaction with other Amateur Radio Operators and Short Wave Listeners. Free technology and related presentations, sponsored construction activities, discounted (and sometimes free) equipment, network of likeminded radio and electronics enthusiasts. Excellent club facilities and environment, ample car parking.

Website: www.nevarc.org.au

Facebook: www.facebook.com/nevicARC/

Postal:
NEVARC Secretary
PO Box 8006
Birallee Park
Wodonga Vic 3690

All editors' comments and other opinions in submitted articles may not always represent the opinions of the committee or the members of NEVARC, but published in spirit, to promote interest and active discussion on club activities and the promotion of Amateur Radio.
Contributions to NEVARC News are always welcome from members.
Email attachments of Word™, Plain Text, Excel™, PDF™ and JPG are all acceptable.
You can post material to the Post Office Box address at the top of this page, or email magazine@nevarc.org.au
Please include a stamped self-addressed envelope if you require your submission notes returned.
Email attachments not to exceed 5 Mb in file size. If you have more than 5 Mb, then send it split, in several emails to us.
Attachments of (or thought to be) executable code or virulently affected emails will not be opened.
Other persons or radio clubs may edit or copy out such as they like from the magazine but a reference to NEVARC News is appreciated, except copyrighted (©) material or as otherwise indicated.
Other articles credited to outside sources should ask for their permission if they are used.
While we strive to be accurate, no responsibility taken for errors, omissions, or other perceived deficiencies, in respect of information contained in technical or other articles.
Any dates, times and locations given for upcoming events please check with a reliable source closer to the event.
This is particularly true for pre-planned outdoor activities affected by adverse weather etc.
The club website http://nevarc.org.au/has current information on planned events and scheduled meeting dates.
You can get the WIA News sent to your inbox each week by simply clicking a link and entering your email address found at www.wia.org.au The links for either text email or MP3 voice files are there as well as Podcasts and Twitter. This WIA service is FREE.